I590 – Python

Spring 2020

Assignment No:	11	Released:	April 5
Title:	Plotting	Due:	April 12

Question 1 (20 points)

A dataset with 25,000 records of heights (in inches) and weights (in pounds) obtained in 1993 by a survey of children from birth to 18 years of age is available at

http://wiki.stat.ucla.edu/socr/index.php/SOCR_Data_Dinov_020108_HeightsWeights

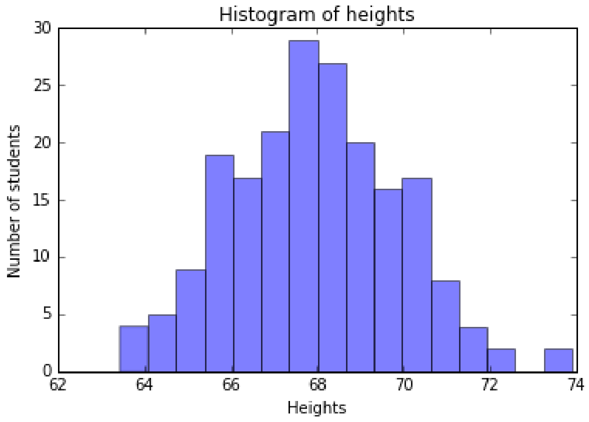
We will use a sample of 200 individuals which is displayed as a table. Manually copy the table, paste it to an Excel sheet and save to an "xlsx" or "csv" file named "heights_weights".

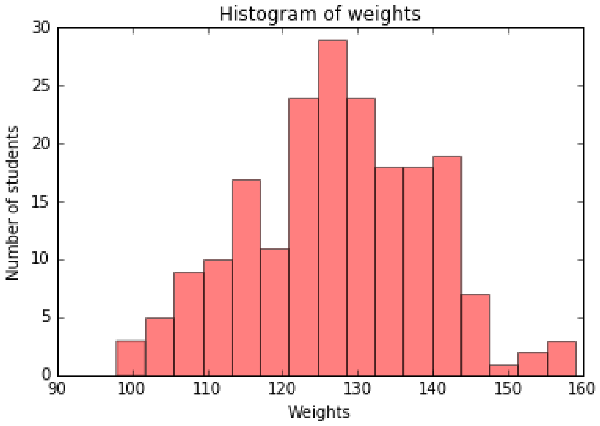
Read data from file "heights_weights" and use *matplotlib* to produce histogram of heights and weights, in separate figures, with 16 bins each.

Include the plot names and labels for the horizontal and vertical axes.

A histogram is a kind of bar plot that gives a discretized display of value frequency. The data points are split into discrete, evenly spaced bins, and the number of data points in each bin is plotted. You can use "hist" method to plot a histogram.

Your histograms should look like the ones displayed below.





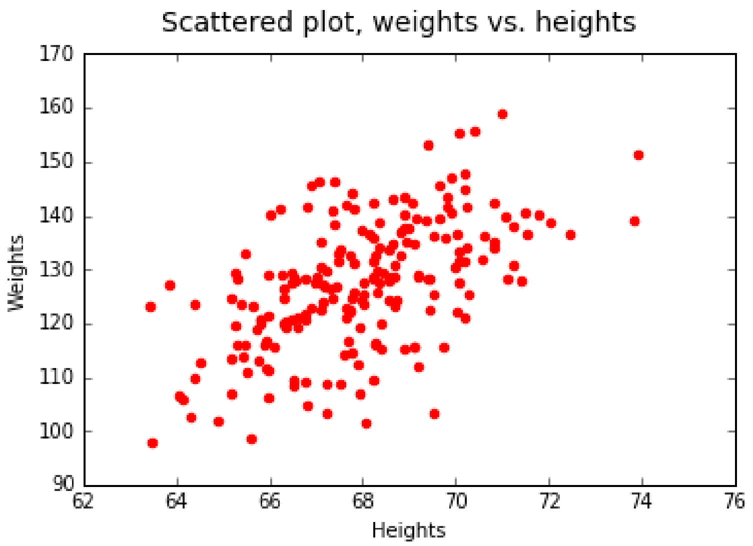
Question 2 (20 points)

Scatter plots are a useful way of examining the relationship between two one-dimensional data series. In this question, use the data that you obtained in Question 1.

Read data from file "heights_weights" and use *matplotlib* to produce a scatter plot of weights versus heights.

Include the name of the plot and labels for the horizontal and vertical axes.

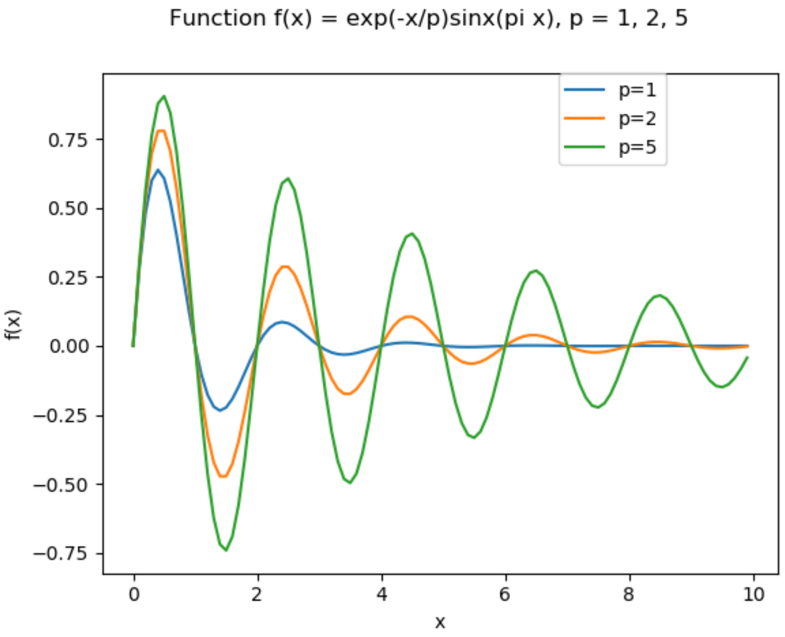
Sample output:



Question 3 (20 points)

Use *matplotlib* to produce a plot of the function *$f(x) = e^{-x/p} \sin(\pi x)$* over the interval *[0, 10]*, for the values of the parameter *p = 1, 2, 5.* Include name of the plot, labels for the x- and y-axes, and a legend that identifies the lines by their color.

Sample output:



Hints:

To plot a function, first create an array of *x* values, for example by using "arrange" function:

```
x = np.arange(0., 10., 0.1)
```

Make sure that you use enough points so that the curve appears smooth.

Create an array of *f(x)* for a specific value of parameter *p*. Use vectorized operations to produce *f(x)*. I wrote a function *f(x, p)* that accepts an array *x* and a scalar *p*, and calculates and returns an array of *f*.

Use "for" loop and plot *f(x)* versus *x* for p = 1, 2, 5:

```
plt.plot(x, f(x, p))
```

Include an optional part to the "plot" function that will produce a label for each line. For example:

```
plt.plot(x, f(x, p), label = str(p))
```

Having plotted all lines, insert legend:

```
plt.legend(bbox_to_anchor=(0.675, 1), loc=2, borderaxespad=0.)
```

For legend guide see:

https://matplotlib.org/users/legend_guide.html

Question 4 (20 points)

The shape of a *limacon* is defined parametrically as:

$r = r_0 + \cos\alpha$

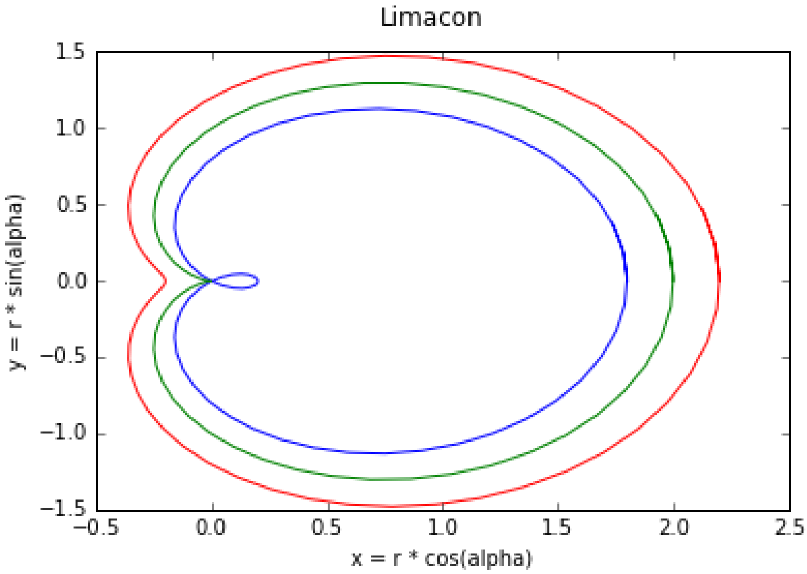
$x = r\cos\alpha$

$y = r\sin\alpha$

When $r_0 = 1$, this curve is called a *cardioid*.

Use *matplotlib* to plot the shape of a *limacon* for $r_0 = 0.8$, $r_0 = 1.0$, and $r_0 = 1.2$. Be sure to use enough points that the curve appears smooth. Include name of the plot, labels for the x- and y-axes, and a legend that identifies the lines by their color.

Sample output (the legend is missing):



Hints:

First, create an array of angle *alpha* values, for example:

```
alpha = np.arange(0., 2.1 * pi, 0.1)
```

Then create arrays of *x* and *y* by using the given formulas and plot *y* vs. *x*. For each line create a label and insert the legend to the plot.

Question 5 (20 points)

A projectile is an object moving only under the force of gravity. In modeling the motion of a projectile we assume no air resistance. In this case, the projectile follows a parabolic path. If an object is launched at an angle θ to the ground with an initial velocity *V*, then the path of the projectile is given by the parametric equations:

$$x = V\,t\cos(\theta)$$

$$y = V\,t\sin(\theta) - (gt^2)/2$$

t represents time (*t = 0* when the object is projected) and *g* is the acceleration due to gravity.

The value of *g* depends upon which system of units we use to measure distance and time. In the metric system we use meters and seconds. In this case, an approximation to *g* is *9.8* meters/second^2 (9.8 m/s^2). The initial velocity is measured in meters per second.

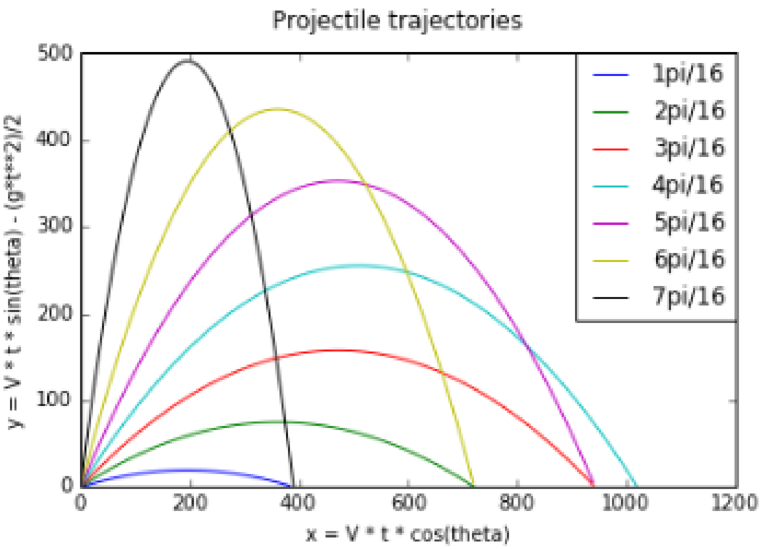
Use *matplotlib* to draw the trajectories of a projectile and determine the angle θ_{max} for which the projectile will reach to longest distance on a horizontal surface. Assume initial velocity *V = 100 m/s* and draw trajectories for θ = (0, π/2), not inclusive, in steps of π/16.

Be sure to use enough points that the curves appear smooth. Include name of the plot, labels for the x- and y-axes, and a legend that identifies the trajectories by their color.

Your program should identify the longest distance reached by the projectile, and print both the max distance and the angle (in radians) that give the longest distance.

Sample output:

```
Angle 4pi/16 gives the longest distance 1020.4 meters.
```



Instructions

- Write your programs in Python 3. Write a separate program for each question.
- Make sure your code is properly formatted, structured and commented. Include a header to program file with your name, date, question number and short program description.
- Do your best to follow the programming style suggested in *Python Enhancement Proposal PEP8* (outlined in a tutorial https://realpython.com/python-pep8/). Follow the naming conventions for variables and functions.
- Make sure that your program compiles. A program that doesn't compile is usually scored 0 points.
- Each program should be stored in a separate Python file. Please name your files such that it starts with the word "question" and ends with the number of the question. All files should be executable python files (they should all end in .py). Acceptable examples include *Question_5.py*, *question_4.py*, *Question3.py*, and *question2.py.*
- A program should run the entire code for a particular question by directly executing it. Include call to `main()` function at the end of your program. Provide any data files that your program needs to execute.
- Please upload to Canvas separate files for each question. If there are five questions, you will be uploading five separate files.
- If you want to provide more instructions for executing your code, upload a 'readme.txt' file, where you may provide additional information.